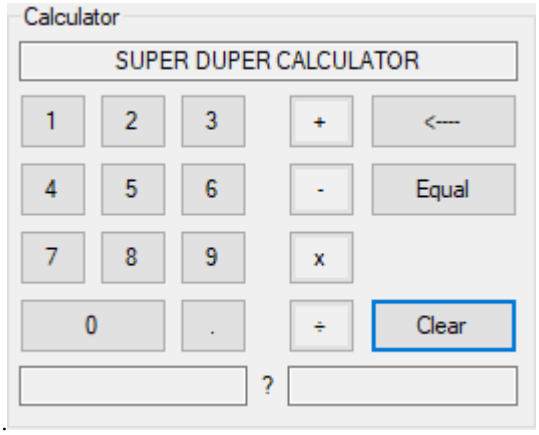
```
Public Class Form1
Private Sub ButtonClickMethod(sender As Object, e As EventArgs) Handles num0.Click, num1.Click,
num2.Click, num3.Click, num4.Click, num5.Click, num6.Click, num7.Click, num8.Click, num9.Click,
opdivision.Click, opmultiply.Click, opdecimal.Click, opclear.Click, opminus.Click, opadd.Click,
opequal.Click
    Dim button As Button = CType(sender, Button)
    If button.Name = "num1" Then
        boxequation1.Text = boxequation1.Text + "1"
    End If
    If button.Name = "num2" Then
        boxequation1.Text = boxequation1.Text + "2"
    End If
    If button.Name = "num3" Then
        boxequation1.Text = boxequation1.Text + "3"
    End If
    If button.Name = "num4" Then
        boxequation1.Text = boxequation1.Text + "4"
    End If
    If button.Name = "num5" Then
        boxequation1.Text = boxequation1.Text + "5"
    End If
    If button.Name = "num6" Then
        boxequation1.Text = boxequation1.Text + "6"
    End If
    If button.Name = "num7" Then
        boxequation1.Text = boxequation1.Text + "7"
    End If
    If button.Name = "num8" Then
        boxequation1.Text = boxequation1.Text + "8"
    End If
    If button.Name = "num9" Then
        boxequation1.Text = boxequation1.Text + "9"
    End If
    If button.Name = "num0" Then
        boxequation1.Text = boxequation1.Text + "0"
    End If
    If button.Name = "opdecimal" Then
        boxequation1.Text = boxequation1.Text + "."
    End If
    If button.Name = "opequal" Then
        Dim equation1 As String = boxequation1.Text
        Dim equation2 As String = boxequation2.Text
        Dim result = New DataTable().Compute(equation1, Nothing)

        boxresult.Text = result
    End If
    If button.Name = "opminus" Then
        boxequation1.Text = boxequation1.Text + "-"
        boxoperator.Text = boxoperator.Text + "-"
    End If
    If button.Name = "opmultiply" Then
        boxequation1.Text = boxequation1.Text + "*"
        boxoperator.Text = boxoperator.Text + "x"
    End If
    If button.Name = "opdivision" Then
        boxequation1.Text = boxequation1.Text + "/"
        boxoperator.Text = boxoperator.Text + "÷"
    End If
    If button.Name = "opadd" Then
        boxequation1.Text = boxequation1.Text + "+"
        boxoperator.Text = boxoperator.Text + "+"
    End If
    If button.Name = "opclear" Then
        boxequation1.Clear()
        boxoperator.Clear()
        boxresult.Clear()
    End If
End Sub
Private Sub opbackspace_Click(sender As Object, e As EventArgs) Handles opbackspace.Click
```

```
    boxequation1.Text = boxequation1.Text.Remove(boxequation1.Text.Count - 1)
End Sub

End Class
```

o learn, after all!

Here what it looks like:



.-.-.-.-.-.-.-.

```
Private Sub New()
    InitializeComponent()

    _currentTextBox = Number1TextBox
    _locked = False
    ResultsTextBox.TextAlign = HorizontalAlignment.Center
    ResultsTextBox.Text = "SUPER DUPER CALCULATOR"
    For Each textBox As System.Windows.Forms.TextBox In {ResultsTextBox, Number1TextBox,
Number2TextBox}
        'This prevent the user from writing in your textboxes instead of using the buttons.
        textBox.ReadOnly = True
    Next

    'I used the .Text and .Tag properties to make the code easier to manage. Observe:
    num0Button.Text = "0"
    num1Button.Text = "1"
    num2Button.Text = "2"
    num3Button.Text = "3"
    '...
    num9Button.Text = "9"
    opAddButton.Text = "+"
    opMinusButton.Text = "-"
    opMultiplyButton.Text = "x"
    opDivideButton.Text = "÷"
    For Each button As Button In {num1Button, num2Button, num3Button, num4Button, num5Button,
num6Button, num7Button, num8Button, num9Button}
        button.Tag = "number"
    Next
    For Each button As Button In {opAddButton, opMinusButton, opMultiplyButton, opDivideButton}
        button.Tag = "operation"
    Next
    num0Button.Tag = "numberZero"
    opDecimalButton.Tag = "decimal"
    opBackspaceButton.Tag = "backspace"
    opEqualButton.Tag = "equal"
End Sub
'Class variables are useful to remember informations which will be useful at several unrelated
places.
'_currentTextBox is a pointer to the TextBox which is currently being filled
Private _currentTextBox As System.Windows.Forms.TextBox = Number1TextBox
```

```
'_Llocked will prevent the user from messing with the calculator after he pressed "equal"
Private _locked As Boolean = False

'I didn't like this idea but I went with it since it was your initial method.
'Notice how shoter this iteration of your idea is. Strive to shorten your code while making your
variable names much easier to read.
'For real, make sure your variables are aptly names. Every control should be identified as such
(the button num0 should be names num0Button for example)
'Code is hard enough to read without making it sibylline
Private Sub ButtonClickMethod(sender As Object, e As EventArgs) Handles num0Button.Click,
num1Button.Click, num2Button.Click, num3Button.Click, num4Button.Click, num5Button.Click,
num6Button.Click, num7Button.Click, num8Button.Click, num9Button.Click, opDivideButton.Click,
opMultiplyButton.Click, opDecimalButton.Click, opClearButton.Click, opMinusButton.Click,
opAddButton.Click, opEqualButton.Click, opBackspaceButton.Click
    'DirectCast is more strick than CType. For this operation, it doesn't really make a
difference, but that's the one you want anyway.
    Dim button As System.Windows.Forms.Button = DirectCast(sender, System.Windows.Forms.Button)

    'The boolean _locked prevent the user from messing with your beautiful calculator.
    If Not _locked Then
        'This is where the .Tag property save lifes. All numbers (but zero) are "number" now.
BAM!
        Select Case button.Tag.ToString
            Case "number"
                _currentTextBox.Text += button.Text
            Case "numberZero"
                'You don't want your clever teacher to try to divide by zero, or create a number
with 5 zeros before the actual number.
                'This is how you can do this.
                If _currentTextBox.Text.Length > 0 Then
                    _currentTextBox.Text += button.Text
                End If
            Case "decimal"
                If _currentTextBox.Text.Length > 0 AndAlso Not _currentTextBox.Text.Contains(".")
Then
                    _currentTextBox.Text += button.Text
                End If
            Case "operation"
                ResetOperationTextBoxes()
                'here's a nifty visual cue about which operation has been selected. It's not
really important.
                button.BackColor = Color.LightSalmon
                'This label will be used to Calculate() the operation. It's important!
                OpLabel.Text = button.Text
            Case "backspace"
                If _currentTextBox.Text.Length > 0 Then
                    _currentTextBox.Text = _currentTextBox.Text.Substring(0,
_currentTextBox.Text.Length - 1)
                End If
            Case "equal"
                'I could have put the code here instead of making another sub, but I didn't.
                'There are two valid reasons for this.
                '1- This code might be useful on it's own later: maybe something else will
Calculate() or LockCalculator or ResetCalculator later.
                '2- try to avoid blobs of code. A big and complex sub is easier to understand
when it's divided between several clean and clear smaller subs.
                '   (of course, you cannot always subdivise Subs. Use common sense)
                Calculate()
        End Select
    End If

    'This check is all alone because I want it to work even if the calculator is locked.
    If button.Tag.ToString = "clear" Then
        ResetCalculator()
    End If
End Sub

Private Sub Calculate()
```

```
    'Always make checks to make sure that things are going the way you think they should be
going.
    'for example, here I want to have 2 numbers and an operation, er else I'll just ignore this
click.
    If Number1TextBox.Text.Length > 0 AndAlso Number2TextBox.Text.Length > 0 And OpLabel.Text <>
"?" Then
        'If the user could mess with the textboxes, i would have to make further checks to avoid
crashes, but I already force the user to use only numbers se we're cool.
        Dim number1 As Double = Convert.ToDouble(Number1TextBox.Text)
        Dim number2 As Double = Convert.ToDouble(Number2TextBox.Text)
        Dim result As Double

        Select Case OpLabel.Text
            Case "+"
                result = number1 + number2
            Case "-"
                result = number1 - number2
            Case "x"
                result = number1 * number2
            Case "÷"
                result = number1 / number2
        End Select

        ResultsTextBox.TextAlign = HorizontalAlignment.Right
        ResultsTextBox.Text = result.ToString
        LockCalculator(True)
    End If
End Sub

'By using a boolean to signify to this Sub if I want to lock or unlock things, I make it easier
to automatize this operation (and I have only one Sub instead of two).
Private Sub LockCalculator(ByVal isLocked As Boolean)
    _locked = isLocked
    'I'm kinda cheating here: I put the calculator inside a GroupBox just so I could be lazier
and do this.
    'Being lazy is great for a coder, as long as you think ahead (and avoid being a nuisance to
yourself when you'll come back to this code and weep).
    For Each control As System.Windows.Forms.Control In CalculatorGroupBox.Controls
        control.Enabled = Not isLocked
    Next
    'Except that I want this button to always be available, so I enable it. This operation
wouldn't be necessary if isLocked is True, but checking for that is longer than just doing it.
    opClearButton.Enabled = True
End Sub

Private Sub ResetCalculator()
    ResetNumberTextBoxes()
    ResetOperationTextBoxes()
    LockCalculator(False)
    _currentTextBox = Number1TextBox
End Sub

Private Sub ResetNumberTextBoxes()
    For Each textBox As System.Windows.Forms.TextBox In {Number1TextBox, Number2TextBox}
        textBox.Text = ""
    Next
    'Mea Culpa: I shouldn't had done a For Each for 2 elements like that. I did it anyway. Please
don't call the Fun Police on me.
    'Also, you can now guess that using For Each loops to set up controls is a work method I
like. Some don't like that.
    'There is about as many ways to code something as there are coders. You have to find your
own, sure, but then you'll get hired somewhere
    'and they will absolutely hate everything that you do that isn't done the way they like it.
And they will be right. On a big project where
    'several coders works, at the same time and for month and years, you have to find a common
ground or else the code will become a nightmare
    'to understand and maintain. Trust me: being able to adapt quickly to other's work
methodology is a GREAT skill in this field of work.
```

```
    ResultsTextBox.TextAlign = HorizontalAlignment.Center
    ResultsTextBox.Text = "SUPER DUPER CALCULATOR"
End Sub

Private Sub ResetOperationTextBoxes()
    For Each button As System.Windows.Forms.Button In {opAddButton, opMinusButton,
opMultiplyButton, opDivideButton}
        button.BackColor = DefaultBackColor
    Next
    OpLabel.Text = "?"
End Sub

'This sub is great. It uses the _currentTextBox as a pointer. Maybe you're not familiar with
pointers, I don't know. The short explanation is this:
'A pointer is like a phone number. You can give your number to several people without "losing"
it. And if they call you and tell you something, they
'are all speaking to the same person.
'_currentTextBox is a pointer which can point toward Number1TextBox or Number2TextBox.
'This Event makes it so when the user click on Number1TextBox or Number2TextBox, the pointer
updates it's "phone number" to the right box,
'so later when we'll write we'll write in the last one which was clicked on.
Private Sub NumberTextBoxes_Click(sender As Object, e As EventArgs) Handles Number1TextBox.Click,
Number2TextBox.Click
    _currentTextBox = DirectCast(sender, System.Windows.Forms.TextBox)
```

```
// Java program to create a simple calculator
// with basic +, -, /, * using java swing elements

import java.awt.event.*;
import javax.swing.*;
import java.awt.*;
class calculator extends JFrame implements ActionListener {
    // create a frame
    static JFrame f;

    // create a textfield
    static JTextField l;

    // store operator and operands
    String s0, s1, s2;

    // default constructor
```

```
    calculator()
    {
        s0 = s1 = s2 = "";
    }

    // main function
    public static void main(String args[])
    {
        // create a frame
        f = new JFrame("calculator");

        try {
            // set look and feel
            UIManager.setLookAndFeel(UIManager.getSystemLookAndFeelClassName());
        }
        catch (Exception e) {
            System.err.println(e.getMessage());
        }

        // create a object of class
        calculator c = new calculator();

        // create a textfield
        l = new JTextField(16);

        // set the textfield to non editable
        l.setEditable(false);

        // create number buttons and some operators
        JButton b0, b1, b2, b3, b4, b5, b6, b7, b8, b9, ba, bs, bd, bm, be, beq,
beq1;

        // create number buttons
        b0 = new JButton("0");
        b1 = new JButton("1");
        b2 = new JButton("2");
        b3 = new JButton("3");
        b4 = new JButton("4");
        b5 = new JButton("5");
        b6 = new JButton("6");
        b7 = new JButton("7");
        b8 = new JButton("8");
        b9 = new JButton("9");

        // equals button
        beq1 = new JButton("=");

        // create operator buttons
        ba = new JButton("+");
        bs = new JButton("-");
        bd = new JButton("/");
        bm = new JButton("*");
        beq = new JButton("C");

        // create . button
        be = new JButton(".");

        // create a panel
        JPanel p = new JPanel();

        // add action listeners
        bm.addActionListener(c);
        bd.addActionListener(c);
        bs.addActionListener(c);
        ba.addActionListener(c);
        b9.addActionListener(c);
        b8.addActionListener(c);
        b7.addActionListener(c);
        b6.addActionListener(c);
```

```
        b5.addActionListener(c);
        b4.addActionListener(c);
        b3.addActionListener(c);
        b2.addActionListener(c);
        b1.addActionListener(c);
        b0.addActionListener(c);
        be.addActionListener(c);
        beq.addActionListener(c);
        beq1.addActionListener(c);

        // add elements to panel
        p.add(l);
        p.add(ba);
        p.add(b1);
        p.add(b2);
        p.add(b3);
        p.add(bs);
        p.add(b4);
        p.add(b5);
        p.add(b6);
        p.add(bm);
        p.add(b7);
        p.add(b8);
        p.add(b9);
        p.add(bd);
        p.add(be);
        p.add(b0);
        p.add(beq);
        p.add(beq1);

        // set Background of panel
        p.setBackground(Color.blue);

        // add panel to frame
        f.add(p);

        f.setSize(200, 220);
        f.show();
    }
    public void actionPerformed(ActionEvent e)
    {
        String s = e.getActionCommand();

        // if the value is a number
        if ((s.charAt(0) >= '0' && s.charAt(0) <= '9') || s.charAt(0) == '.') {
            // if operand is present then add to second no
            if (!s1.equals(""))
                s2 = s2 + s;
            else
                s0 = s0 + s;

            // set the value of text
            l.setText(s0 + s1 + s2);
        }
        else if (s.charAt(0) == 'C') {
            // clear the one letter
            s0 = s1 = s2 = "";

            // set the value of text
            l.setText(s0 + s1 + s2);
        }
        else if (s.charAt(0) == '=') {

            double te;

            // store the value in 1st
            if (s1.equals("+"))
                te = (Double.parseDouble(s0) + Double.parseDouble(s2));
            else if (s1.equals("-"))
```

```
                te = (Double.parseDouble(s0) - Double.parseDouble(s2));
            else if (s1.equals("/"))
                te = (Double.parseDouble(s0) / Double.parseDouble(s2));
            else
                te = (Double.parseDouble(s0) * Double.parseDouble(s2));

            // set the value of text
            l.setText(s0 + s1 + s2 + "=" + te);

            // convert it to string
            s0 = Double.toString(te);

            s1 = s2 = "";
        }
        else {
            // if there was no operand
            if (s1.equals("") || s2.equals(""))
                s1 = s;
            // else evaluate
            else {
                double te;

                // store the value in 1st
                if (s1.equals("+"))
                    te = (Double.parseDouble(s0) + Double.parseDouble(s2));
                else if (s1.equals("-"))
                    te = (Double.parseDouble(s0) - Double.parseDouble(s2));
                else if (s1.equals("/"))
                    te = (Double.parseDouble(s0) / Double.parseDouble(s2));
                else
                    te = (Double.parseDouble(s0) * Double.parseDouble(s2));

                // convert it to string
                s0 = Double.toString(te);

                // place the operator
                s1 = s;

                // make the operand blank
                s2 = "";
            }

            // set the value of text
            l.setText(s0 + s1 + s2);
        }
    }
}
```

Output :

a = 10, b = 20, c = 30

For AND operator:

Condition 1: c > a
Condition 2: c > b

Output: True [Both Conditions are true]

For OR Operator:

```
Condition 1: c > a
Condition 2: c > b
```

Output: True [One of the Condition if true]

For NOT Operator:

```
Condition 1: c > a
Condition 2: c > b

a = 10, b = 20, c = 20

condition1: a < b
condition2: b == c

if(condition1 && condition2)
d = a + b + c

// Since both the conditions are true
d = 50.
```

Example

```
// Java code to illustrate

// logical AND operator



import java.io.*;



class Logical {

    public static void main(String[] args)

    {

        // initializing variables

        int a = 10, b = 20, c = 20, d = 0;



        // Displaying a, b, c

        System.out.println("Var1 = " + a);

        System.out.println("Var2 = " + b);

        System.out.println("Var3 = " + c);



        // using logical AND to verify

        // two constraints

        if ((a < b) && (b == c)) {

            d = a + b + c;
```

```
            System.out.println("The sum is: " + d);

        }

        else

            System.out.println("False
conditions");

    }

}
```

Output

```
Var1 = 10
Var2 = 20
Var3 = 20
The sum is: 50
```

Now in the below example, we can see the short-circuiting effect. Here when the execution reaches to if statement, the first condition inside the if statement is false and so the second condition is never checked. Thus the ++b(pre-increment of b) never happens and b remains unchanged.

Example:

```
import java.io.*;



class shortCircuiting {

    public static void main(String[] args)

    {



        // initializing variables

        int a = 10, b = 20, c = 15;



        // displaying b

        System.out.println("Value of b : " + b);



        // Using logical AND

        // Short-Circuiting effect as the first condition
is

        // false so the second condition is never reached

        // and so ++b(pre increment) doesn't take place and

        // value of b remains unchanged
```

```
        if ((a > c) && (++b > c)) {

            System.out.println("Inside if block");

        }


        // displaying b

        System.out.println("Value of b : " + b);

    }

}
```

Output:

The output of AND Operator

2. Logical 'OR' Operator (||)

This operator returns true when one of the two conditions under consideration is satisfied or is true. If even one of the two yields true, the operator results true. To make the result false, both the constraints need to return false.

Syntax:

```
condition1 || condition2
```

Example:


```
a = 10, b = 20, c = 20

condition1: a < b
condition2: b > c

if(condition1 || condition2)
d = a + b + c

// Since one of the condition is true
d = 50.
// Java code to illustrate

// logical OR operator



import java.io.*;



class Logical {
```

```
    public static void main(String[] args)
    {
        // initializing variables
        int a = 10, b = 1, c = 10, d = 30;

        // Displaying a, b, c
        System.out.println("Var1 = " + a);
        System.out.println("Var2 = " + b);
        System.out.println("Var3 = " + c);
        System.out.println("Var4 = " + d);

        // using logical OR to verify
        // two constraints
        if (a > b || c == d)
            System.out.println(
                "One or both + the conditions are
true");
        else
            System.out.println(
                "Both the + conditions are false");
    }
}
```

Output

```
Var1 = 10
Var2 = 1
Var3 = 10
Var4 = 30
One or both + the conditions are true
```

Now in the below example, we can see the short-circuiting effect for OR operator. Here when the execution reaches to if statement, the first condition inside the if statement is true and so the second condition is never checked. Thus the ++b (pre-increment of b) never happens and b remains unchanged.

Example

```
import java.io.*;
```

```
class ShortCircuitingInOR {

    public static void main (String[] args) {


      // initializing variables

        int a = 10, b = 20, c = 15;


        // displaying b

        System.out.println("Value of b: " +b);


        // Using logical OR

        // Short-circuiting effect as the first condition is
true

        // so the second condition is never reached

        // and so ++b (pre-increment) doesn't take place and

        // value of b remains unchanged

        if((a < c) || (++b < c))

            System.out.println("Inside if");


        // displaying b

        System.out.println("Value of b: " +b);


    }

}
```

Output

```
Value of b: 20
Inside if
Value of b: 20
```

3. Logical 'NOT' Operator (!)

Unlike the previous two, this is a unary operator and returns true when the condition under consideration is not satisfied or is a false condition. Basically, if the condition is false, the operation returns true and when the condition is true, the operation returns false.

Syntax:

```
!(condition)
```

Example:

```
a = 10, b = 20

!(a<b) // returns false
!(a>b) // returns true
// Java code to illustrate

// logical NOT operator

import java.io.*;



class Logical {

    public static void main(String[] args)

    {

        // initializing variables

        int a = 10, b = 1;



        // Displaying a, b, c

        System.out.println("Var1 = " + a);

        System.out.println("Var2 = " + b);



        // Using logical NOT operator

        System.out.println("!(a < b) = " + !(a <
b));

        System.out.println("!(a > b) = " + !(a >
b));

    }

}
```

Output

```
Var1 = 10
Var2 = 1
!(a < b) = true
!(a > b) = false
boolean a = true;
 boolean b = false;
```

program –

```
public class LogicalOperators {

    public static void main(String[] args) {

        boolean a = true;

        boolean b = false;
```

```
        System.out.println("a: " + a);

        System.out.println("b: " + b);

        System.out.println("a && b: " + (a &&
b));

        System.out.println("a || b: " + (a ||
b));

        System.out.println("!a: " + !a);

        System.out.println("!b: " + !b);

    }

}
```

Output

```
a: true
b: false
a && b: false
a || b: true
!a: false
!b: true
```

Calculator in Java with Source Code

Calculator in Java with Source Code: We can develop calculator in java with the help of AWT/Swing with event handling. Let's see the code of creating calculator in java.

```
1.  /*******************************************
2.  Save this file as MyCalculator.java
3.  to compile it use
4.      javac MyCalculator.java
5.  to use the calcuator do this
6.      java MyCalculator
7.
8.  *******************************************/
9.  import java.awt.*;
10. import java.awt.event.*;
11. /*******************************************/
12.
13. public class MyCalculator extends Frame
14. {
15.
16. public boolean setClear=true;
17. double number, memValue;
18. char op;
19.
20. String digitButtonText[] = {"7", "8", "9", "4", "5", "6", "1", "2", "3", "0", "+/-", "." };
21. String operatorButtonText[] = {"/", "sqrt", "*", "%", "-", "1/X", "+", "=" };
22. String memoryButtonText[] = {"MC", "MR", "MS", "M+" };
23. String specialButtonText[] = {"Backspc", "C", "CE" };
24.
25. MyDigitButton digitButton[]=new MyDigitButton[digitButtonText.length];
26. MyOperatorButton operatorButton[]=new MyOperatorButton[operatorButtonText.length];
27. MyMemoryButton memoryButton[]=new MyMemoryButton[memoryButtonText.length];
28. MySpecialButton specialButton[]=new MySpecialButton[specialButtonText.length];
29.
30. Label displayLabel=new Label("0",Label.RIGHT);
```

```
Label memLabel=new Label(" ",Label.RIGHT);

final int FRAME_WIDTH=325,FRAME_HEIGHT=325;
final int HEIGHT=30, WIDTH=30, H_SPACE=10,V_SPACE=10;
final int TOPX=30, TOPY=50;
//////////////////////////
MyCalculator(String frameText)//constructor
{
super(frameText);

int tempX=TOPX, y=TOPY;
displayLabel.setBounds(tempX,y,240,HEIGHT);
displayLabel.setBackground(Color.BLUE);
displayLabel.setForeground(Color.WHITE);
add(displayLabel);

memLabel.setBounds(TOPX,  TOPY+HEIGHT+ V_SPACE,WIDTH, HEIGHT);
add(memLabel);

// set Co-ordinates for Memory Buttons
tempX=TOPX;
y=TOPY+2*(HEIGHT+V_SPACE);
for(int i=0; i<memoryButton.length; i++)
{
memoryButton[i]=new MyMemoryButton(tempX,y,WIDTH,HEIGHT,memoryButtonText[i], this);
memoryButton[i].setForeground(Color.RED);
y+=HEIGHT+V_SPACE;
}

//set Co-ordinates for Special Buttons
tempX=TOPX+1*(WIDTH+H_SPACE); y=TOPY+1*(HEIGHT+V_SPACE);
for(int i=0;i<specialButton.length;i++)
{
specialButton[i]=new MySpecialButton(tempX,y,WIDTH*2,HEIGHT,specialButtonText[i], this);
specialButton[i].setForeground(Color.RED);
tempX=tempX+2*WIDTH+H_SPACE;
}

//set Co-ordinates for Digit Buttons
int digitX=TOPX+WIDTH+H_SPACE;
int digitY=TOPY+2*(HEIGHT+V_SPACE);
tempX=digitX;  y=digitY;
for(int i=0;i<digitButton.length;i++)
{
digitButton[i]=new MyDigitButton(tempX,y,WIDTH,HEIGHT,digitButtonText[i], this);
digitButton[i].setForeground(Color.BLUE);
tempX+=WIDTH+H_SPACE;
if((i+1)%3==0){tempX=digitX; y+=HEIGHT+V_SPACE;}
}

//set Co-ordinates for Operator Buttons
int opsX=digitX+2*(WIDTH+H_SPACE)+H_SPACE;
int opsY=digitY;
tempX=opsX;  y=opsY;
for(int i=0;i<operatorButton.length;i++)
{
tempX+=WIDTH+H_SPACE;
operatorButton[i]=new MyOperatorButton(tempX,y,WIDTH,HEIGHT,operatorButtonText[i], this);
operatorButton[i].setForeground(Color.RED);
if((i+1)%2==0){tempX=opsX; y+=HEIGHT+V_SPACE;}
}

addWindowListener(new WindowAdapter()
{
public void windowClosing(WindowEvent ev)
{System.exit(0);}
});

setLayout(null);
```

```
100.setSize(FRAME_WIDTH,FRAME_HEIGHT);
101.setVisible(true);
102.}
103.////////////////////////////////////
104.static String getFormattedText(double temp)
105.{
106.String resText=""+temp;
107.if(resText.lastIndexOf(".0")>0)
108.    resText=resText.substring(0,resText.length()-2);
109.return resText;
110.}
111./////////////////////////////////////////
112.public static void main(String []args)
113.{
114.new MyCalculator("Calculator - JavaTpoint");
115.}
116.}
117.
118./*****************************************/
119.
120.class MyDigitButton extends Button implements ActionListener
121.{
122.MyCalculator cl;
123.
124./////////////////////////////////////////
125.MyDigitButton(int x,int y, int width,int height,String cap, MyCalculator clc)
126.{
127.super(cap);
128.setBounds(x,y,width,height);
129.this.cl=clc;
130.this.cl.add(this);
131.addActionListener(this);
132.}
133./////////////////////////////////////////////////
134.static boolean isInString(String s, char ch)
135.{
136.for(int i=0; i<s.length();i++) if(s.charAt(i)==ch) return true;
137.return false;
138.}
139./////////////////////////////////////////////////
140.public void actionPerformed(ActionEvent ev)
141.{
142.String tempText=((MyDigitButton)ev.getSource()).getLabel();
143.
144.if(tempText.equals("."))
145.{
146. if(cl.setClear)
147.    {cl.displayLabel.setText("0.");cl.setClear=false;}
148. else if(!isInString(cl.displayLabel.getText(),'.'))
149.    cl.displayLabel.setText(cl.displayLabel.getText()+".");
150. return;
151.}
152.
153.int index=0;
154.try{
155.        index=Integer.parseInt(tempText);
156.    }catch(NumberFormatException e){return;}
157.
158.if (index==0 && cl.displayLabel.getText().equals("0")) return;
159.
160.if(cl.setClear)
161.            {cl.displayLabel.setText(""+index);cl.setClear=false;}
162.else
163.    cl.displayLabel.setText(cl.displayLabel.getText()+index);
164.}//actionPerformed
165.}//class defination
166.
167./*****************************************/
168.
```

```
class MyOperatorButton extends Button implements ActionListener
{
MyCalculator cl;

MyOperatorButton(int x,int y, int width,int height,String cap, MyCalculator clc)
{
super(cap);
setBounds(x,y,width,height);
this.cl=clc;
this.cl.add(this);
addActionListener(this);
}
//////////////////////
public void actionPerformed(ActionEvent ev)
{
String opText=((MyOperatorButton)ev.getSource()).getLabel();

cl.setClear=true;
double temp=Double.parseDouble(cl.displayLabel.getText());

if(opText.equals("1/x"))
    {
    try
        {double tempd=1/(double)temp;
        cl.displayLabel.setText(MyCalculator.getFormattedText(tempd));}
    catch(ArithmeticException excp)
                        {cl.displayLabel.setText("Divide by 0.");}
    return;
    }
if(opText.equals("sqrt"))
    {
    try
        {double tempd=Math.sqrt(temp);
        cl.displayLabel.setText(MyCalculator.getFormattedText(tempd));}
            catch(ArithmeticException excp)
                    {cl.displayLabel.setText("Divide by 0.");}
    return;
    }
if(!opText.equals("="))
    {
    cl.number=temp;
    cl.op=opText.charAt(0);
    return;
    }
// process = button pressed
switch(cl.op)
{
case '+':
    temp+=cl.number;break;
case '-':
    temp=cl.number-temp;break;
case '*':
    temp*=cl.number;break;
case '%':
    try{temp=cl.number%temp;}
    catch(ArithmeticException excp)
        {cl.displayLabel.setText("Divide by 0."); return;}
    break;
case '/':
    try{temp=cl.number/temp;}
        catch(ArithmeticException excp)
                {cl.displayLabel.setText("Divide by 0."); return;}
    break;
}//switch

cl.displayLabel.setText(MyCalculator.getFormattedText(temp));
//cl.number=temp;
}//actionPerformed
}//class
```

```
238.
239./****************************************/
240.
241.class MyMemoryButton extends Button implements ActionListener
242.{
243.MyCalculator cl;
244.
245.//////////////////////////////////
246.MyMemoryButton(int x,int y, int width,int height,String cap, MyCalculator clc)
247.{
248.super(cap);
249.setBounds(x,y,width,height);
250.this.cl=clc;
251.this.cl.add(this);
252.addActionListener(this);
253.}
254.////////////////////////////////////////////////
255.public void actionPerformed(ActionEvent ev)
256.{
257.char memop=((MyMemoryButton)ev.getSource()).getLabel().charAt(1);
258.
259.cl.setClear=true;
260.double temp=Double.parseDouble(cl.displayLabel.getText());
261.
262.switch(memop)
263.{
264.case 'C':
265.    cl.memLabel.setText(" ");cl.memValue=0.0;break;
266.case 'R':
267.    cl.displayLabel.setText(MyCalculator.getFormattedText(cl.memValue));break;
268.case 'S':
269.    cl.memValue=0.0;
270.case '+':
271.    cl.memValue+=Double.parseDouble(cl.displayLabel.getText());
272.    if(cl.displayLabel.getText().equals("0") || cl.displayLabel.getText().equals("0.0")  )
273.        cl.memLabel.setText(" ");
274.    else
275.        cl.memLabel.setText("M");
276.    break;
277.}//switch
278.}//actionPerformed
279.}//class
280.
281./****************************************/
282.
283.class MySpecialButton extends Button implements ActionListener
284.{
285.MyCalculator cl;
286.
287.MySpecialButton(int x,int y, int width,int height,String cap, MyCalculator clc)
288.{
289.super(cap);
290.setBounds(x,y,width,height);
291.this.cl=clc;
292.this.cl.add(this);
293.addActionListener(this);
294.}
295.//////////////////////
296.static String backSpace(String s)
297.{
298.String Res="";
299.for(int i=0; i<s.length()-1; i++) Res+=s.charAt(i);
300.return Res;
301.}
302.
303.////////////////////////////////////////////////////////
304.public void actionPerformed(ActionEvent ev)
305.{
306.String opText=((MySpecialButton)ev.getSource()).getLabel();
```

```
307.//check for backspace button
308.if(opText.equals("Backspc"))
309.{
310.String tempText=backSpace(cl.displayLabel.getText());
311.if(tempText.equals(""))
312.    cl.displayLabel.setText("0");
313.else
314.    cl.displayLabel.setText(tempText);
315.return;
316.}
317.//check for "C" button i.e. Reset
318.if(opText.equals("C"))
319.{
320.cl.number=0.0; cl.op=' '; cl.memValue=0.0;
321.cl.memLabel.setText(" ");
322.}
323.
324.//it must be CE button pressed
325.cl.displayLabel.setText("0");cl.setClear=true;
326.}//actionPerformed
327.}//class
328.
329./**********************************************
330.Features not implemented and few bugs
331.
332.i)  No coding done for "+/-" button.
333.ii) Menubar is not included.
334.iii)Not for Scientific calculation
335.iv)Some of the computation may lead to unexpected result
336.   due to the representation of Floating point numbers in computer
337.   is an approximation to the given value that can be stored
338.   physically in memory.
339.**********************************************/
```

download this example

```
 eate a simple Program to build the Calculator in JavaScript using with HTML and CSS web languages. -->
 <!DOCTYPE html>
 <html lang = "en">
 <head>
 <title> JavaScript Calculator </title>

 <style>
 h1 {
     text-align: center;
     padding: 23px;
     background-color: skyblue;
     color: white;
     }

 #clear{
 width: 270px;
 border: 3px solid gray;
     border-radius: 3px;
     padding: 20px;
     background-color: red;
 }

 .formstyle
 {
 width: 300px;
 height: 530px;
 margin: auto;
 border: 3px solid skyblue;
 border-radius: 5px;
 padding: 20px;
 }


```

```
 input
 {
 width: 20px;
 background-color: green;
 color: white;
 border: 3px solid gray;
     border-radius: 5px;
     padding: 26px;
     margin: 5px;
     font-size: 15px;
 }


 #calc{
 width: 250px;
 border: 5px solid black;
     border-radius: 3px;
     padding: 20px;
     margin: auto;
 }

 </style>

 </head>
 <body>
 <h1> Calculator Program in JavaScript </h1>
 <div class= "formstyle">
 <form name = "form1">

     <!-- This input box shows the button pressed by the user in calculator. -->
   <input id = "calc" type ="text" name = "answer"> <br> <br>
   <!-- Display the calculator button on the screen. -->
   <!-- onclick() function display the number prsses by the user. -->
   <input type = "button" value = "1" onclick = "form1.answer.value += '1' ">
   <input type = "button" value = "2" onclick = "form1.answer.value += '2' ">
   <input type = "button" value = "3" onclick = "form1.answer.value += '3' ">
    <input type = "button" value = "+" onclick = "form1.answer.value += '+' ">
   <br> <br>

   <input type = "button" value = "4" onclick = "form1.answer.value += '4' ">
   <input type = "button" value = "5" onclick = "form1.answer.value += '5' ">
   <input type = "button" value = "6" onclick = "form1.answer.value += '6' ">
   <input type = "button" value = "-" onclick = "form1.answer.value += '-' ">
   <br> <br>

   <input type = "button" value = "7" onclick = "form1.answer.value += '7' ">
   <input type = "button" value = "8" onclick = "form1.answer.value += '8' ">
   <input type = "button" value = "9" onclick = "form1.answer.value += '9' ">
   <input type = "button" value = "*" onclick = "form1.answer.value += '*' ">
   <br> <br>


   <input type = "button" value = "/" onclick = "form1.answer.value += '/' ">
   <input type = "button" value = "0" onclick = "form1.answer.value += '0' ">
     <input type = "button" value = "." onclick = "form1.answer.value += '.' ">
     <!-- When we click on the '=' button, the onclick() shows the sum results on the calculator screen. -->
   <input type = "button" value = "=" onclick = "form1.answer.value = eval(form1.answer.value) ">
   <br>
   <!-- Display the Cancel button and erase all data entered by the user. -->
   <input type = "button" value = "Clear All" onclick = "form1.answer.value = ' ' " id= "clear" >
   <br>

 </form>
 </div>
 </body>
 </html>
```

- o
-
-

Java project| Create Page for ATM

// ATM Application using gui in java Source Code

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import java.applet.*;
public class Banking2 extends JPanel implements ActionListener
{
```

```
// Create ATM GUI Java Frame
JFrame f2;

 JLabel l1,l2,l3,l4,l5,l6,user,l7,l8; // declared label to display text
 JTextField t1,t2,t5,t7;  // declare area to type some text
JTextArea t4,t3;          // for multiple text like description
 JPasswordField pw;  // for password field
 JButton b1,b2,ok;      // for button
 JRadioButton r1,r2,r3;  // ratio button
JTextArea output;

Banking2()
{
// this is use to display text on frame
f2=new JFrame("Create your a/c");

// this is use to display text label on the screen
l1=new JLabel("Name:");
l2=new JLabel("Surname");
l3=new JLabel("Discription");
l4=new JLabel("Date of Birth");
l5=new JLabel("Gender");
l6=new JLabel("address");
user=new JLabel("User Name:");
 l7=new JLabel("password");
l8=new JLabel("phone no");

// this area is use to write text as well as to enter password and
//  number
t1=new JTextField();
t2=new JTextField();
t3=new JTextArea();
t4=new JTextArea();
t5=new JTextField();
pw=new JPasswordField();
t7=new JTextField();

output=new JTextArea();  //for output

// this are is use to display image on screen
```

```
ImageIcon bg=new ImageIcon(“atm.png”);
ImageIcon bg1=new ImageIcon(“ok.png”);
ImageIcon bg2=new ImageIcon(“cancel.png”);

//ImageIcon bg3=new ImageIcon(“ok.png”);
JLabel bgi=new JLabel(” “,bg,JLabel.CENTER);

// for selecting gender using the radio button
ButtonGroup o=new ButtonGroup();
r1=new JRadioButton(“male”,false);
r2=new JRadioButton(“female”,false);
r3=new JRadioButton(“other”,false);

// this for checking the condition
 JCheckBox cond=new JCheckBox(“I accept all term and condition”);

b1=new JButton(bg1);
b2=new JButton(bg2);

//ok=new JButton(bg3);
Choice c=new Choice();
Choice c2=new Choice();
Choice c3=new Choice();

// about size , layout , color of frame
f2.setVisible(true);
f2.setSize(1000,1000);
f2.setLayout(null);
f2.setBackground(Color.YELLOW);
f2.setForeground(Color.BLUE);
bgi.setBounds(500,0,500,500);

// location of the object in the frame

cond.setBounds(10,400,400,20);
output.setBounds(500,0,500,500);
output.setBackground(Color.yellow);
output.setForeground(Color.red);

l1.setBounds(10,10,100,20);
t1.setBounds(110,10,200,20);

l2.setBounds(10,50,100,20);
t2.setBounds(110,50,200,20);

l3.setBounds(10,90,100,20);
t3.setBounds(110,90,200,30);

l4.setBounds(10,130,100,20);
c.setBounds(110,130,100,20);
c2.setBounds(210,130,100,20);
c3.setBounds(310,130,100,20);

l5.setBounds(10,170,100,20);
r1.setBounds(110,170,100,20);
r2.setBounds(210,170,100,20);
r3.setBounds(310,170,100,20);

l6.setBounds(10,210,100,20);
t4.setBounds(110,210,200,20);
t4.setEditable(false);
user.setBounds(10,250,100,20);
t5.setBounds(110,250,200,20);

l7.setBounds(10,290,100,20);
pw.setBounds(110,290,200,20);
```

```
l8.setBounds(10,330,100,20);
t7.setBounds(110,330,200,20);

b1.setBounds(30,500,500,100);
b2.setBounds(600,500,500,100);
//ok.setBounds(360,500,300,100);

// adding all the component inside the frame
f2.add(l1);
f2.add(t1);
f2.add(l2);
f2.add(t2);
f2.add(l3);
f2.add(t3);
f2.add(l4);
f2.add(c);
f2.add(c2);
f2.add(c3);
f2.add(bgi);

.. to select  day of birthday
c.add("1");
c.add("2");
c.add("3");
c.add("4");
c.add("5");
c.add("6");
c.add("7");
c.add("8");
c.add("9");
c.add("10");
c.add("11");
c.add("12");
c.add("13");
c.add("14");
c.add("15");
c.add("16");
c.add("17");
c.add("18");
c.add("19");
c.add("20");
c.add("21");
c.add("22");
c.add("23");
c.add("24");
c.add("25");
c.add("26");
c.add("27");
c.add("28");
c.add("29");
c.add("30");
c.add("31");

// this for month

c2.add("1");
c2.add("2");
c2.add("3");
c2.add("4");
c2.add("5");
c2.add("6");
c2.add("7");
c2.add("8");
c2.add("9");
c2.add("10");
c2.add("11");
c2.add("12");
```

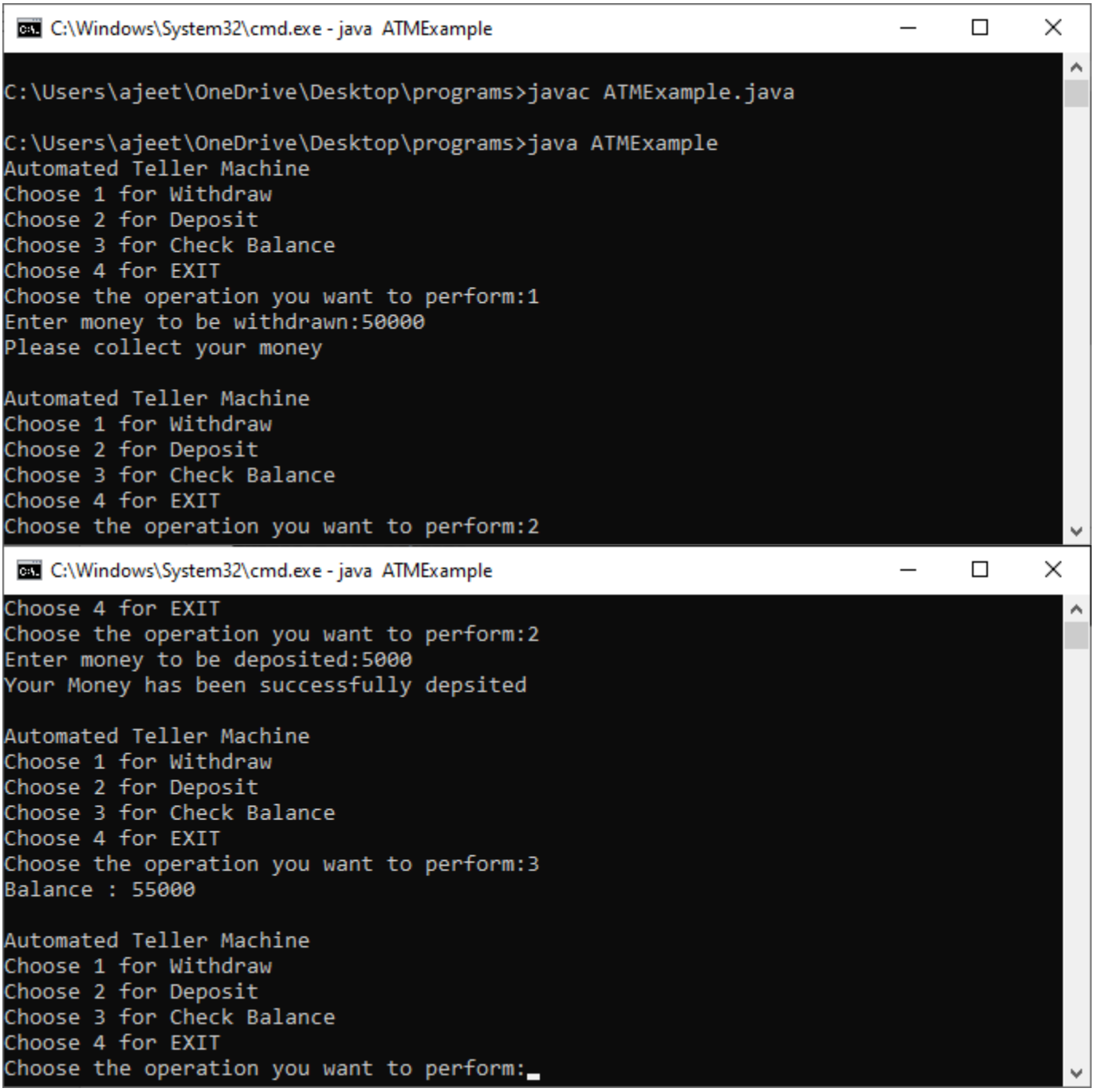
```
// this is for year
c3.add(“2000”);
c3.add(“2001”);
c3.add(“2002”);
c3.add(“2003”);
c3.add(“2004”);
c3.add(“2005”);
c3.add(“2006”);
c3.add(“2007”);
c3.add(“2008”);
c3.add(“2009”);
c3.add(“2010”);
c3.add(“2011”);
c3.add(“2012”);
c3.add(“2013”);
c3.add(“2014”);
c3.add(“2015”);
c3.add(“2016”);
c3.add(“2017”);
c3.add(“2018”);
c3.add(“2019”);

f2.add(l5);
o.add(r1);
o.add(r2);
o.add(r3);
f2.add(r1);
f2.add(r2);
f2.add(r3);

f2.add(l6);
f2.add(t4);
f2.add(user);
f2.add(t5);
f2.add(l7);
 f2.add(pw);
f2.add(l8);
 f2.add(t7);

f2.add(b1);
//f2.add(ok);
f2.add(b2);

f2.add(output);
f2.add(cond);

// adding listener to the button

cond.addActionListener(this);
b1.addActionListener(this);
b2.addActionListener(this);
ok.addActionListener(this);
}
public void actionPerformed(ActionEvent e)
{
String name=t1.getText();
String surname=t2.getText();
String discription=t3.getText();
//String day=c.getSelectedItem();
//String month=c2.getSelectedItem();
//String years=c3.getSelectedItem();
String gender=”male”;
if(r2.isSelected()==true)
gender=”female”;
if(r3.isSelected()==true)
gender=”other”;
String address=t4.getText();
String users=t5.getText();
String pass=pw.getText();
```

```
String phone=t7.getText();

if(e.getSource().equals(b1))
{
if(t1.getText().isEmpty()||(t2.getText().isEmpty())||(t3.getText().isEmpty())||(t4.getText().isEmpty())||
(t5.getText().isEmpty())||(pw.getText().isEmpty())||(t7.getText().isEmpty())||(r1.isSelected())||(r2.isSelected())||
(r3.isSelected()))
{
JOptionPane.showMessageDialog(null,”Data missing”);
}
else
{
JOptionPane.showMessageDialog(null,”Data completed”);
}
output.setText(“name:-  “+name+”nSurname:-  “+surname+”nDiscription:-  “+discription+ “nGender:-
“+gender+”nAddress:- “+address+ “nUser Name:- “+users+ “nPass word:- “+pass+”nMobile No:-“+phone);


//FileReader fr=new FileReader(“abc.txt”);
}
if(e.getSource().equals(b2))
{
f2.setVisible(false);
new Banking();
}
}
public static void  main(String s[])
{
new Banking2();
}
}
```

```
1.  //import required classes and packages
2.  import java.util.Scanner;
3.
4.  //create ATMExample class to implement the ATM functionality
5.  public class ATMExample
6.  {
7.      //main method starts
8.      public static void main(String args[] )
9.      {
10.         //declare and initialize balance, withdraw, and deposit
11.         int balance = 100000, withdraw, deposit;
12.
13.         //create scanner class object to get choice of user
14.         Scanner sc = new Scanner(System.in);
15.
16.         while(true)
17.         {
18.             System.out.println("Automated Teller Machine");
19.             System.out.println("Choose 1 for Withdraw");
20.             System.out.println("Choose 2 for Deposit");
21.             System.out.println("Choose 3 for Check Balance");
22.             System.out.println("Choose 4 for EXIT");
23.             System.out.print("Choose the operation you want to perform:");
24.
25.             //get choice from user
26.             int choice = sc.nextInt();
27.             switch(choice)
28.             {
29.                 case 1:
30.         System.out.print("Enter money to be withdrawn:");
31.
32.         //get the withdrawl money from user
33.         withdraw = sc.nextInt();
```

```
34.
35.         //check whether the balance is greater than or equal to the withdrawal amount
36.         if(balance >= withdraw)
37.         {
38.            //remove the withdrawl amount from the total balance
39.            balance = balance - withdraw;
40.            System.out.println("Please collect your money");
41.         }
42.         else
43.         {
44.            //show custom error message
45.            System.out.println("Insufficient Balance");
46.         }
47.         System.out.println("");
48.         break;
49.
50.               case 2:
51.
52.         System.out.print("Enter money to be deposited:");
53.
54.         //get deposite amount from te user
55.         deposit = sc.nextInt();
56.
57.         //add the deposit amount to the total balanace
58.         balance = balance + deposit;
59.         System.out.println("Your Money has been successfully depsited");
60.         System.out.println("");
61.         break;
62.
63.               case 3:
64.         //displaying the total balance of the user
65.         System.out.println("Balance : "+balance);
66.         System.out.println("");
67.         break;
68.
69.               case 4:
70.         //exit from the menu
71.         System.exit(0);
72.            }
73.         }
74.      }
75.  }
```

Output:




```
package atm;

import java.sql.SQLException;

public class Main {
	public static void main(String[] args) throws InterruptedException, SQLException {
		Login login = new Login();
		login.loginView();
	}
}
package atm;

import java.awt.Font;
```

```
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.MouseAdapter;
import java.awt.event.MouseEvent;

import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JTextField;

public class Login extends Commons{
	public void loginView() {
		Commons common = new Commons();
		JFrame frame = (JFrame)common.Frame();
		Font txt = new Font("", Font.BOLD, 15);
		Pin pin = new Pin();


		//---------------CARDNUMBER----------------
		JLabel card = new JLabel("ENTER YOUR CARD NUMBER");
		card.setBounds(50, 270, 250, 20);
		card.setFont(txt);
		JTextField cardNumber = new JTextField();
		cardNumber.setBounds(50, 300, 500, 35);
		cardNumber.setFont(txt);
		frame.add(cardNumber);
		frame.add(card);
		//-----------------------------------------


		//----------------ADMIN--------------------
		JLabel admin = new JLabel("ADMIN LOGIN >");
		admin.setBounds(0, 500, 570, 30);
		admin.setHorizontalAlignment(JLabel.RIGHT);
		admin.setFont(txt);
		frame.add(admin);
		admin.addMouseListener(new MouseAdapter() {
			public void mousePressed(MouseEvent e) {
				pin.pinView("admin");
				frame.dispose();
			}
		});
		//-----------------------------------------


		//-----------------BUTTON-----------------
		JButton cont = new JButton("COUNTINUE");
		cont.setBounds(200, 400, 200, 50);
		cont.setFont(new Font("Rockwell", Font.BOLD, 25));
		frame.add(cont);
		cont.addActionListener(new ActionListener() {
			@Override
			public void actionPerformed(ActionEvent e) {
				if(cardNumber.getText().length() == 16) {
					pin.pinView(cardNumber.getText());
					frame.dispose();
				}
				else {
					Fail fail = new Fail();
					fail.failView("WRONG CARD NUMBER!!!");
					frame.dispose();
				}
			}

		});
		//-----------------------------------------
		frame.setVisible(true);
	}
}ackage atm;

import java.awt.Font;
```

```
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.sql.SQLException;
import java.util.Random;

import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JTextField;

public class AddUser {

	JTextField pinField, atmField;
	Random random = new Random();

	public void addView() throws SQLException {
		Commons commons = new Commons();
		JFrame frame = (JFrame) commons.Frame();
		Font txt = new Font("", Font.BOLD, 20);
		SQLManage manage = new SQLManage();
		Success success = new Success();

		//--------------NAME--------------------
		JLabel name = new JLabel("Name : ");
		name.setBounds(50, 200, 100, 25);
		name.setFont(txt);
		JTextField nmField = new JTextField();
		nmField.setBounds(50, 230, 500, 30);
		frame.add(nmField);
		frame.add(name);
		//----------------------------------------

		//-------------ATMNUMBER-----------------
		JLabel atmno = new JLabel("ATM Card Number : ");
		atmno.setBounds(50, 300, 500, 25);
		atmno.setFont(txt);
		atmField = new JTextField();
		atmField.setBounds(50, 330, 500, 30);
		atmField.setEditable(false);
		frame.add(atmField);
		frame.add(atmno);
		//----------------------------------------

		//-------------ATMPIN-----------------
		JLabel atmpin = new JLabel("ATM Card PIN : ");
		atmpin.setBounds(50, 400, 500, 25);
		atmpin.setFont(txt);
		pinField = new JTextField();
		pinField.setBounds(50, 430, 200, 30);
		pinField.setEditable(false);
		frame.add(pinField);
		frame.add(atmpin);
		//----------------------------------------

		//-------------BALANCE-----------------
		JLabel bal = new JLabel("BALANCE : ");
		bal.setBounds(350, 400, 500, 25);
		bal.setFont(txt);
		JTextField balField = new JTextField();
		balField.setBounds(350, 430, 200, 30);
		frame.add(balField);
		frame.add(bal);
		//----------------------------------------

		//--------------AUTOGENERATION----------------
		auto();
		//----------------------------------------------
```

```
		//---------------SUBMIT-------------------
		JButton sbmt = new JButton("SUBMIT");
		sbmt.setBounds(200, 500, 200, 50);
		frame.add(sbmt);
		sbmt.addActionListener(new ActionListener() {
			@Override
			public void actionPerformed(ActionEvent e) {
				if(!nmField.getText().equals("")) {
					if(balField.getText().equals(""))
						balField.setText("0");
					try {
						manage.adding(atmField.getText(), pinField.getText(),
nmField.getText(), balField.getText());
					} catch (SQLException e1) {
						e1.printStackTrace();
					}
					success.detailView(atmField.getText(), pinField.getText());
					balField.setText("");
					nmField.setText("");
					auto();
				}
			}
		});
		//-----------------------------------------


		frame.setVisible(true);
	}

	public void auto() {
		String str = "";
		for(int i=0; i<16; i++) {
			str += random.nextInt(9 - 0 + 1) + 0;
		}
		atmField.setText(str);
		str = "";
		for(int i=0; i<4; i++) {
			str += random.nextInt(9 - 0 + 1) + 0;
		}
		pinField.setText(str);
	}
}
```

Employee Management System Project in Java

Mohsin Shaikh August 28, 2022

Introduction

We are going to develop an Employee Management System Project in Java. This project is great for those who are at an intermediate level and want to advance their coding skills. We will be creating a GUI interface using the swing package.

This will be a GUI-based program with MySQL as a database. Administrators will be able to perform the following functionalities:

- Login
- View all employees
- Create employee
- Edit employee
- Delete employee
- Save data

Let's get started!

Employee Management System Project in Java: Project Overview

Project Name:	**Employee Management System Project in Java**
Abstract	It's a GUI-based project used with the Swing module to organize all the elements that work

	under library management.
Language/s Used:	Java
IDE	IntelliJ Idea Professional(Recommended)
Java version (Recommended):	**Java SE 18.0. 2.1**
Database:	MySQL
Type:	Desktop Application
Recommended for	Final Year Students

I am using IntelliJ as my IDE. You can use any. You must have java JDK installed on your system. The first step will be to create a new project. Name it as you wish. In the src folder create an EmployeeManagementSystem.java file. We will be writing our code here.

Before we start

To connect the system with the database you will need to follow certain steps.

- Have Java JDK already installed and an IDE like IntelliJ or Eclipse
- Install MySQL on your pc.
- Download MySQL connector from here.
- In IntelliJ, under your project expand external libraries and right-click on junit4, and select Open Library Settings. Select the libraries tab and click on the + button. Browse your jar file downloaded from the above step and click on it. This will add dependency to your project. The steps will differ if you are using a different IDE.

MySQL steps

Create database

```
create database ems;
```

Select the database

```
use ems;
```

Create employee table

```
create table employee (
  id int primary key,
  name varchar(25),
  gender varchar(10),
  phoneNum varchar(13),
  email varchar(25),
  designation varchar(20),
  salary double
);
```

Create admin table

```
create table admin (
  username varchar(25) primary key,
  password varchar(25)
);
```

Insert some values in the admin table for login

```
insert into admin values
("admin","admin123"),
("admin2","0000");
```

Insert values into employees (optional)

```
insert into employee values
(142,"Jon Snow","Male","8454562158","jon@gmail.com","Associate",30000.0),
(127,"Robb Stark","Male","7654812345","robb@gmail.com","Manager",80000.0);
```

Employee Management System Project in Java

1. Importing required libraries

```
import java.awt.*;
import java.awt.event.KeyAdapter;
import java.awt.event.KeyEvent;
import java.sql.*;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;
import javax.swing.*;
import javax.swing.border.EmptyBorder;
```

2. Create a connection with SQL in Java

Give the name of your database, username, and password in the below fields.

```
// gets the required driver
Class.forName("com.mysql.cj.jdbc.Driver");
// creates connection with sql
String dbName = "ems";
String db_username = "root";
String db_password = "master";
Connection con= DriverManager.getConnection(
        "jdbc:mysql://localhost:3306/"+dbName, db_username, db_password);
```

Create an instance of the employee management system and pass to it the connection object so that it can later be used.

```
EmployeeManagementSystem ems = new EmployeeManagementSystem(con);
```

3. Admin Login for employee management system in java

Users should be able to log in with their credentials. We are also going to show popups on invalid usernames or passwords.

First, we will create the login screen by using JFrame and add all components to it.

```
ems.loginFrame = new JFrame();

JLabel usernameLabel = new JLabel("Username");
usernameLabel.setBounds(200,150,220,50);
usernameLabel.setFont(new Font("Times New Roman", Font.PLAIN, 30));
ems.loginFrame.add(usernameLabel);

JTextField usernameField = new JTextField();
usernameField.setFont(new Font("Times New Roman", Font.PLAIN, 20));
usernameField.setBounds(450,150,420,50);
ems.loginFrame.add(usernameField);

JLabel passwordLabel = new JLabel("Password");
passwordLabel.setFont(new Font("Times New Roman", Font.PLAIN, 30));
passwordLabel.setBounds(200,250,220,50);
ems.loginFrame.add(passwordLabel);

JPasswordField passwordField = new JPasswordField();
passwordField.setFont(new Font("Times New Roman", Font.PLAIN, 20));
passwordField.setBounds(450,250,420,50);
ems.loginFrame.add(passwordField);
```

```
JButton submitButton=new JButton("Submit");
submitButton.setFont(new Font("Times New Roman", Font.PLAIN, 30));
submitButton.setBounds(450,400,250, 50);
ems.loginFrame.add(submitButton);

ems.loginFrame.setSize(1100,750);
ems.loginFrame.setLayout(null);
ems.loginFrame.setVisible(true);
ems.loginFrame.setDefaultCloseOperation(WindowConstants.EXIT_ON_CLOSE);
```

We get the below screen with this code

Now let’s add functionalities to it

We will create an action listener on the submit button. This will allow us to get the user input and validate it for authentication

```
submitButton.addActionListener(actionEvent -> {
    username = usernameField.getText();
    password = String.valueOf(passwordField.getPassword());

    try {
        boolean auth = ems.login(username, password);
        if(auth){
            ems.loginFrame.dispose();
            ems.mainMenu();
        }

    } catch (SQLException throwables) {
        throwables.printStackTrace();
    }
});
```

4. Login screen for employee management system java project

Now let’s write the login functionality

```
public boolean login(String username, String password) throws SQLException {
        Statement statement = this.con.createStatement();
        String q = String.format("select password from admin where username = '%s';", username);
        ResultSet resultSet = statement.executeQuery(q);
//        gets us password if the username exists
        if(resultSet.next()){
//            compare password with user input
            if(resultSet.getString(1).equals(password)){
                return true;
            }
            else {
                JFrame popup = new JFrame("Invalid password");
                JLabel popupMsg = new JLabel("The password you entered is invalid.");
                popupMsg.setBounds(20,10,300,50);
                popupMsg.setFont(new Font("Times New Roman", Font.PLAIN, 20));
                popup.add(popupMsg);

                JButton button = new JButton("OK");
                button.setBounds(120,60,70,20);
                button.setFont(new Font("Times New Roman", Font.PLAIN, 20));
                button.addActionListener(actionEvent2 -> {
                    popup.dispose();
                });
                popup.add(button);

                popup.setLayout(null);
                popup.setSize(350, 150);
                popup.setVisible(true);
                return false;
            }
        }
        else {
            JFrame popup = new JFrame("Invalid username");
            JLabel popupMsg = new JLabel("The username you entered does not exist.");
            popupMsg.setBounds(20,10,500,50);
            popupMsg.setFont(new Font("Times New Roman", Font.PLAIN, 20));
            popup.add(popupMsg);

            JButton button = new JButton("OK");
            button.setBounds(170,60,70,20);
            button.setFont(new Font("Times New Roman", Font.PLAIN, 20));
            button.addActionListener(actionEvent2 -> {
```

```
                popup.dispose();
            });
            popup.add(button);

            popup.setLayout(null);
            popup.setSize(450, 150);
            popup.setVisible(true);
            return false;
        }
    }
```

Java Program to Calculate Electricity Bill

Write a Java Program to Calculate Electricity Bill using Else If statement with examples.

Java Program to Calculate Electricity Bill using Else If

This Java program asks the user to enter the units consumed. Next, it calculates the Total Electricity bill. This approach is useful if the Electricity board is charging different tariffs for different units. For this example, we are using the Else If Statement.

TIP: The Java Else If statement checks the first condition if it is TRUE, it executes the statements inside that block. If the condition is FALSE, it checks the Next one (Else If condition) and so on.

```
import java.util.Scanner;

public class ElectricityBill1 {
          private static Scanner sc;
          public static void main(String[] args)
          {
                    int Units;
                    double Amount, Sur_Charge, Total_Amount;
                    sc = new Scanner(System.in);

                    System.out.print(" Please Enter the Units that you Consumed  : ");
                    Units = sc.nextInt();

                    if (Units < 50)
                    {
                   Amount = Units * 2.60;
                              Sur_Charge = 25;
                    }
                    else if (Units <= 100)
                    {
                              // For the First Fifty Units Charge = 130 (50 * 2.60)
                              // Next, we are removing those 50 units from total units
                              Amount = 130 + ((Units - 50 ) * 3.25);
                              Sur_Charge = 35;
                    }
                    else if (Units <= 200)
                    {
                              // First Fifty Units charge = 130, and 50 - 100 is 162.50 (50 *
3.25)
                              // Next, we are removing those 100 units from total units
                              Amount = 130 + 162.50 + ((Units - 100 ) * 5.26);
                              Sur_Charge = 45;
                    }
                    else
                    {
                              /* First Fifty Units = 130, 50 - 100 is 162.50,
                               and 100 - 200 is 526 (100 * 5.65)
```

```
                        Next, we are removing those 200 units from total units */
                        Amount = 130 + 162.50 + 526 + ((Units - 200 ) * 7.75);
                        Sur_Charge = 55;
                }

                Total_Amount = Amount + Sur_Charge;
                System.out.println("\n Electricity Bill  =  " + Total_Amount);
        }

}
```

Java Program to find Electricity Bill using Method

This program to calculate electricity bills is the same as the first example. But we separated the logic and placed it in a separate method.

```
import java.util.Scanner;
```

```
public class Example2 {
	private static Scanner sc;
	public static void main(String[] args)
	{
		int Units;
		double Total_Amount;
		sc = new Scanner(System.in);

		System.out.print(" Please Enter the Units that you Consumed  : ");
		Units = sc.nextInt();

		Total_Amount = ElecBill(Units);
		System.out.println("\n Electricity Bill  =  " + Total_Amount);
	}
	public static double ElecBill(int Units)
	{
		double Amount, Sur_Charge, Total_Amount;
		if (Units < 50)
		{
		Amount = Units * 2.60;
			Sur_Charge = 25;
		}
		else if (Units <= 100)
		{
			Amount = 130 + ((Units - 50 ) * 3.25);
			Sur_Charge = 35;
		}
		else if (Units <= 200)
		{
			Amount = 130 + 162.50 + ((Units - 100 ) * 5.26);
			Sur_Charge = 45;
		}
		else
		{
			Amount = 130 + 162.50 + 526 + ((Units - 200 ) * 7.75);
			Sur_Charge = 55;
		}

		Total_Amount = Amount + Sur_Charge;
		return Total_Amount;
	}
}
 Please Enter the Units that you Consumed  : 95

 Electricity Bill  =  311.25
```

Let me try a different value.


```
 Please Enter the Units that you Consumed  : 450

 Electricity Bill  =  2811.0
```

Java Program to find Electricity Bill Example

This type of Java program is useful if the board has uniform rates. Something like: if you consume between 300 and 500 units, then charges are fixed as 7.75 Rupees for Units, etc.

```
import java.util.Scanner;

public class Example3 {
	private static Scanner sc;
	public static void main(String[] args)
	{
```

```
                    int Units;
                    double Amount, Sur_Charge, Total_Amount;
                    sc = new Scanner(System.in);

                    System.out.print(" Please Enter the Units that you Consumed  : ");
                    Units = sc.nextInt();

                    if (Units > 500)
                    {
                              Amount = Units * 9.65;
                              Sur_Charge = 85;
                    } // Otherwise (fails), Compiler will move to Next Else If block
                    else if (Units >= 300)
                    {
                              Amount = Units * 7.75;
                              Sur_Charge = 75;
                    }
                    else if (Units >= 200)
                    {
                              Amount = Units * 5.26;
                              Sur_Charge = 55;
                    }
                    else if (Units >= 100)
                    {
                              Amount = Units * 3.76;
                              Sur_Charge = 35;
                    } // Otherwise (fails), Compiler will move to Else block
                    else
                    {
                              Amount = Units * 2.25;
                              Sur_Charge = 25;
                    }
                    Total_Amount = Amount + Sur_Charge;
                    System.out.println("\n Electricity Bill  =  " + Total_Amount);
          }
}
 Please Enter the Units that you Consumed  : 750

 Electricity Bill  =  7322.5
```

Let me try a different value.

```
 Please Enter the Units that you Consumed  : 250

 Electricity Bill  =  1370.0
C++ implementation to calculate the
// electricity bill
#include<bits/stdc++.h>
using namespace std;

// Function to calculate the
// electricity bill
int calculateBill(int units)
{

    // Condition to find the charges
    // bar in which the units consumed
    // is fall
    if (units <= 100)
    {
        return units * 10;
    }
    else if (units <= 200)
    {
        return (100 * 10) +
               (units - 100) * 15;
    }
    else if (units <= 300)
```

```
    {
        return (100 * 10) +
               (100 * 15) +
               (units - 200) * 20;
    }
    else if (units > 300)
    {
        return (100 * 10) +
               (100 * 15) +
               (100 * 20) +
               (units - 300) * 25;
    }
    return 0;
}

// Driver Code
int main()
{
    int units = 250;
    cout << calculateBill(units);
}

// This code is contributed by spp____
```

put: U = 250
Output: 3500
Explanation:
Charge for the first 100 units – 10*100 = 1000
Charge for the 100 to 200 units – 15*100 = 1500
Charge for the 200 to 250 units – 20*50 = 1000
Total Electricity Bill = 1000 + 1500 + 1000 = 3500
Input: U = 95
Output: 950
Explanation:
Charge for the first 100 units – 10*95 = 950
Total Electricity Bill = 950

B

```
' Multiline syntax:
If condition [ Then ]
    [ statements ]
[ ElseIf elseifcondition [ Then ]
    [ elseifstatements ] ]
[ Else
    [ elsestatements ] ]
End If

' Single-line syntax:
If condition Then [ statements ] [ Else [ elsestatements ] ]
```

he following example illustrates the use of the multiline syntax of the `If...Then...Else` statement.

VB

```
'Create a Random object to seed our starting value
Dim randomizer As New Random()
'set our variable
Dim count As Integer = randomizer.Next(0, 5)

Dim message As String

'If count is zero, output will be no items
If count = 0 Then
    message = "There are no items."
    'If count is 1, output will be "There is 1 item.".
ElseIf count = 1 Then
    message = "There is 1 item."
    'If count is greater than 1, output will be "There are {count} items.", where {count} is
replaced by the value of count.
```

```
Else
    message = $"There are {count} items."
End If

Console.WriteLine(message)

'This example displays output like the following:
' There are 4 items.
```

Nested syntax example

The following example contains nested `If...Then...Else` statements.

VB

```
Public Sub Main()
    ' Run the function as part of the WriteLine output.
    Console.WriteLine("Time Check is " & CheckIfTime() & ".")
End Sub

Private Function CheckIfTime() As Boolean
    ' Determine the current day of week and hour of day.
    Dim dayW As DayOfWeek = DateTime.Now.DayOfWeek
    Dim hour As Integer = DateTime.Now.Hour

    ' Return True if Wednesday from 2 to 3:59 P.M.,
    ' or if Thursday from noon to 12:59 P.M.
    If dayW = DayOfWeek.Wednesday Then
        If hour = 14 Or hour = 15 Then
            Return True
        Else
            Return False
        End If
    ElseIf dayW = DayOfWeek.Thursday Then
        If hour = 12 Then
            Return True
        Else
            Return False
        End If
    Else
        Return False
    End If
End Function

'This example displays output like the following:
'Time Check is False.

Private Sub SingleLine()

    'Create a Random object to seed our starting values
    Dim randomizer As New Random()

    Dim A As Integer = randomizer.Next(10, 20)
    Dim B As Integer = randomizer.Next(0, 20)
    Dim C As Integer = randomizer.Next(0, 5)

    'Let's display the initial values for comparison
    Console.WriteLine($"A value before If: {A}")
    Console.WriteLine($"B value before If: {B}")
    Console.WriteLine($"C value before If: {C}")

    ' If A > 10, execute the three colon-separated statements in the order
    ' that they appear
    If A > 10 Then A = A + 1 : B = B + A : C = C + B

    'If the condition is true, the values will be different
    Console.WriteLine($"A value after If: {A}")
    Console.WriteLine($"B value after If: {B}")
```

```
    Console.WriteLine($"C value after If: {C}")

End Sub

'This example displays output like the following:
'A value before If: 11
'B value before If: 6
'C value before If: 3
'A value after If: 12
Option Compare Binary

Console.WriteLine("A" < "a")
' Output: True
Option Compare Text

Console.WriteLine("A" = "a")
' Output: True
```

```
         Declare a one-dimensional Integer array and a Single
' variable.
Dim philaMint(5) As Integer
Dim x As Single
x! = 10.0
philaMint%(0) = 3                ' Assigns an Integer value
philaMint%(1) = x                ' Converts Single 10.0 to Integer 10
Print DataType(philaMint%(0)); DataType(philaMint%(1))
' Output:
' 2  2
' Both values are Integers.
eclare marCell and perDue as Integer variables.
' The phrase As Integer declares marCell as an Integer
' variable. The data type suffix % declares perDue as an
' Integer variable.
Dim marCell As Integer, perDue%
Print TypeName(marCell), TypeName(perDue%)    ' Prints INTEGER INTEGER
Dim marCell% As Integer
' Error, because the Dim statement attempts to declare
' the Integer variable marCell using both the data type
' suffix character for Integer, and the data type name
' Integer. The declaration should include one or the
' other, but not both.
' A data type suffix character is optional in references to a
' declared variable.
' Declare marCell as an Integer variable.
Dim marCell As Integer
' Use the data type suffix character in a reference to marCell.
marCell% = 1
' Refer to marCell without using the suffix character.
Print marCell                        ' Prints 1
```

The Visual Basic printer object greatly simplifies sending output to a printer. The following is a very simple example of creating a PDF and specifying the output file name.

```
Private Sub Command1_Click()

 'set the output file name

 SaveSetting "Dane Prairie Systems", "Win2PDF", "PDFFileName", "c:\TEST.PDF"
```

```
 'output some text

 Printer.Print "hello world!"



 'save the PDF file

 Printer.EndDoc

End Sub
```

Win2PDF supports all of the standard Printer object methods and properties. A more advanced example requires some support routines for writing a Dword value to the registry. The following code must be added to the declarations section of the sample form.

```
Option Explicit



Private Const REG_DWORD As Long = 4

Private Const HKEY_CURRENT_USER = &H80000001

Private Const KEY_ALL_ACCESS = &H3F



Private Declare Function RegCloseKey Lib "advapi32.dll" _

        (ByVal hKey As Long) As Long

Private Declare Function RegOpenKeyEx Lib "advapi32.dll" Alias _

  "RegOpenKeyExA" (ByVal hKey As Long, ByVal lpSubKey As String, _

  ByVal ulOptions As Long, ByVal samDesired As Long, phkResult As _

  Long) As Long

Private Declare Function RegSetValueExLong Lib "advapi32.dll" Alias _

  "RegSetValueExA" (ByVal hKey As Long, ByVal lpValueName As String, _

  ByVal Reserved As Long, ByVal dwType As Long, lpValue As Long, _

  ByVal cbData As Long) As Long
```

```
Private Sub SaveWin2PDFDword(sValueName As String, lValueSetting As Long)

 Dim lRetVal As Long     'result of the SetValueExLong function

 Dim hKey As Long       'handle of open key



 lRetVal = RegOpenKeyEx(HKEY_CURRENT_USER, "Software\VB and VBA Program Settings\Dane Prairie
Systems\Win2PDF", 0, _

          KEY_ALL_ACCESS, hKey)

 lRetVal = RegSetValueExLong(hKey, sValueName, 0&, _

          REG_DWORD, lValueSetting, 4)

 RegCloseKey (hKey)

End Sub
```

Now we can use the subroutine to save a Dword value to the registry and enable 128 bit encryption. The encryption options are only valid for Win2PDF Pro.

```
Private Sub Command1_Click()

 'set the output file name

 SaveSetting "Dane Prairie Systems", "Win2PDF", "PDFFileName", "c:\TEST.PDF"



 'enable 128 bit encryption (Win2PDF Pro only) and automatic url detection

 SaveWin2PDFDword "file options", &H30



 'set a master password

 SaveSetting "Dane Prairie Systems", "Win2PDF", "master password", "abracadabra"



 'set the PDF document title

 SaveSetting "Dane Prairie Systems", "Win2PDF", "PDFTitle", "Hello World sample"



 'set landscape orientation
```

```
 Printer.Orientation = vbPRORLandscape


 'output some text
 Printer.Print "hello world!"
 Printer.Print "www.win2pdf.com"


 'save the PDF file
 Printer.EndDoc
End Sub
```

The following example demonstrates how to append to an existing PDF file.

```
Private Sub Command2_Click()


 'set the output file name to create original PDF file
 SaveSetting "Dane Prairie Systems", "Win2PDF", "PDFFileName", "c:\TEST.PDF"


 'output some text
 Printer.Print "hello world!"


 'save the PDF file
 Printer.EndDoc




 'Append new text to original PDF file
 'assign original PDF file
 SaveSetting "Dane Prairie Systems", "Win2PDF", "PDFAppendFile", "c:\TEST.PDF"
```

```
 'set new output file to maintain the original PDF file

 SaveSetting "Dane Prairie Systems", "Win2PDF", "PDFFileName", "c:\AppendedTest.PDF"



 Printer.Print "2nd page added"

 Printer.EndDoc



End Sub
```

The following example enables the "Send PDF" option, sets the email subject, sets the email address, sets the email body text, and then creates the PDF file:

```
 'enable sending the PDF file

 'replace the file options with &H421 to email with no user interaction using the Mail Helper
application

 SaveWin2PDFDword "file options", &H21



 'set the email subject (optional)

 SaveSetting "Dane Prairie Systems", "Win2PDF", "PDFMailSubject", "Test Subject"



 'set the recipient email address (optional)

 SaveSetting "Dane Prairie Systems", "Win2PDF", "PDFMailRecipients", billg@microsoft.com



 'set the email body text (optional)

 SaveSetting "Dane Prairie Systems", "Win2PDF", "PDFMailNote", "The requested PDF file is
attached."



 'set the PDF file name

 SaveSetting "Dane Prairie Systems", "Win2PDF", "PDFFileName", "c:\\test.pdf"
```

```
'output some text

Printer.Print "hello world!"

Printer.Print www.win2pdf.com



'save the PDF file

Printer.EndDoc
```

Upgrading the Printer Object

Last Updated on Sat, 04 Feb 2023 | Upgrading Visual Basic

In Visual Basic 6.0, the Printer object enables applications to communicate with a system printer. The Printer object contains functions for printing text, lines, and images. However, the printing model in the .NET Framework is quite different from that in Visual Basic 6.0. The Printer object has become obsolete. Fortunately, the functionality of the Printer object can be achieved with the classes that are provided in the System.Drawing.Printing namespace, particularly the PrintDocument class.

There are two main options to upgrade the Visual Basic 6.0 printing functionality. The first option is to replace the Printer object members with equivalent functionality provided by the PrintDocument class. Typically, this requires reimplementation of the functionality; this will be demonstrated in the next code example. The second option is to create your own Printer class in Visual Basic .NET based on the .NET PrintDocument class. This approach will allow you to mask the .NET PrintDocument class functionality with the names provided in the Visual Basic 6.0 Printer object. This second option will also be demonstrated later in this section.

Applying the first approach will require reimplementation of the printing code. The following Visual Basic 6.0 code example prints four lines of text with different font settings:

Private Sub Form_Load() With Printer

Printer.Print "Normal Line" .FontBold = True Printer.Print "Bold Line" .FontItalic = True

Printer.Print "Bold and Italic Line" .FontBold = False .FontItalic = False Printer.Print "Second Normal Line" .EndDoc End With End Sub

This code can be reimplemented in Visual Basic .NET using methods of the PrintDocument class, as shown here.

Dim WithEvents prn As New Printing.PrintDocument

Private Sub Form1_Load(ByVal eventSender As System.Object, _ ByVal eventArgs As System.EventArgs) Handles MyBase.Load prn.Print() End Sub

Private Sub prn_PrintPage(ByVal sender As Object, _

ByVal e As System.Drawing.Printing.PrintPageEventArgs) Handles prn.PrintPage Dim currFont As Font

Dim yPos As Integer yPos = 1

With e.Graphics currFont = New Font("Arial", 10, FontStyle.Regular) .DrawString("Normal Line", currFont, Brushes.Black, 1, yPos) yPos = yPos + currFont.GetHeight currFont = New Font("Arial", 10, FontStyle.Bold) .DrawString("Bold Line", currFont, Brushes.Black, 1, yPos) yPos = yPos + currFont.GetHeight currFont = New Font("Arial", 10, FontStyle.Bold Or FontStyle.Italic) .DrawString("Bold and Italic Line", currFont,

Brushes.Black, 1, yPos) yPos = yPos + currFont.GetHeight currFont = New Font("Arial", 10, FontStyle.Regular) .DrawString("Second Normal Line", currFont, Brushes.Black, 1, yPos) End With End Sub

Notice how the structure of the code has changed and how the quantity of drawing methods has increased. The .NET Framework provides more control and power over the drawing operations, but from the upgrade point of view, it has a learning curve, and the manual reimplementation of all the printing functionality can consume extensive resources.

For more information about the equivalent functionality for the Printer object members, see Tables 7.3 and 7.4 at the end of this section. For further guidance on printing from Visual Basic .NET with the PrintDocument component, see "Printing with the PrintDocument Component" in on MSDN.

The second upgrade approach involves the creation of your own Printer class. This allows you to consolidate several drawing methods and collections of graphical objects (such as Circle or Line) and to store the coordinates that will be used to print these objects when the Print method or the EndDoc method is called. When the Print method of the PrintDocument class is called, the PrintPage event is raised. This event can be used to signal the application to draw all the objects and text stored in the collections of your PrinterClass. The following Visual Basic .NET code provides a small demonstration of how this Printer class can be developed.

Imports Microsoft.VisualBasic.Compatibility ' Imports Microsoft.VisualBasic. Public Class PrinterClass

Public Enum FillStyleConstants As Short vbFSSolid = 0 vbFSTransparent = 1 End Enum

' Scale mode Constants Public Enum ScaleModeConstants As Short vbTwips = 1

vbPixels = 3 End Enum

Private Structure LineInfo

Dim pen As System.Drawing.Pen Dim pl, p2 As Drawing.Point End Structure

Private Structure RectangleInfo Dim pen As System.Drawing.Pen Dim rec As Drawing.Rectangle Dim FillStyle As FillStyleConstants Dim FillColor As System.Drawing.Color End Structure

Private Structure PageInfo

Public Lines() As LineInfo Public Circ1es() As RectangleInfo End Structure

Private Pages() As PageInfo Private PageIndex = 0

Private WithEvents InnerPrinter As New System.Drawing.Printing.PrintDocument Private objPen As New System.Drawing.Pen(System.Drawing.Brushes.Black) Private intCurrentX As Double = 20 Private intCurrentY As Double = 20

Public ScaleMode As ScaleModeConstants = ScaleModeConstants.vbTwips Public FillColor As Drawing.Color

Public FillStyle As FillStyleConstants = FillStyleConstants.vbFSTransparent

Public Sub New() ReDim Pages(0) Pages(0) = New PageInfo End Sub

' Terminates a print operation sent to the Printer object, ' releasing the document to the print device or spooler. Public Sub EndDoc() Dim j As Integer For j = 0 To Pages.Length - l PageIndex = j InnerPrinter.Print()

Next End Sub

Public Sub Circ1e(ByVa1 P As Drawing.Point, ByVal radius As Double, _ Optional ByVal [Step] As Boolean = False) Circ1e(P, radius, objPen.Color, [Step]) End Sub

```
Public Sub Circ1e(ByVa1 P As Drawing.Point, ByVal radius As Double, _ ByVal Color As Drawing.Color, _ Optional ByVal [Step] As Boolean = False)

Dim diameter As Double

P.X = P.X + CurrentX P.Y = P.Y + CurrentY End If diameter = radius * 2

' Moves the CurrentX and CurrentY properties CurrentX = P.X + radius CurrentY = P.Y + radius

P.X = ConvertToPixelsX(P.X) P.Y = ConvertToPixelsY(P.Y) diameter = ConvertToPixelsX(diameter)

If IsNothing(Pages(PageIndex).Circles) Then ReDim Preserve Pages(PageIndex).Circles(0)

Else

ReDim Preserve

Pages(PageIndex).Circles(Pages(PageIndex).Circles.Length) End If

Pages(PageIndex).Circles(Pages(PageIndex).Circles.Length

New Drawing.Rectangle(P.X, P.Y, diameter, diameter) Pages(PageIndex).Circles(Pages(PageIndex).Circles.Length
Pages(PageIndex).Circles(Pages(PageIndex).Circles.Length

FillColor

Pages(PageIndex).Circles(Pages(PageIndex).Circles.Length FillStyle

End Sub

Private Sub Printer_PrintPage(ByVal sender As Object, _

ByVal e As System.Drawing.Printing.PrintPageEventArgs) Handles InnerPrinter.PrintPage Dim i As Integer ' Draw
all circles

If IsNothing(Pages(PageIndex).Circles) = False Then For i = 0 To Pages(PageIndex).Circles.Length - 1 Dim x, y As
Integer e.Graphics.DrawEllipse(Pages(PageIndex).Circles(i).pen,

Pages(PageIndex).Circles(i).rec) If Pages(PageIndex).Circles(i).FillStyle = _ FillStyleConstants.vbFSSolid Then
e.Graphics.FillEllipse( _

New Drawing.SolidBrush( _ Pages(PageIndex).Circles(i).FillColor), _ Pages(PageIndex).Circles(i).rec)

End If

Next End If End Sub

Public Property CurrentX() As Double Get

Return ConvertToPixelsX(intCurrentX) End Get

Set(ByVal Value As Double)

intCurrentX = ConvertToPixelsX(Value) End Set

End Property
```

Public Property CurrentY() As Double Get

Return ConvertToPixelsY(intCurrentY) End Get

Set(ByVal Value As Double)

intCurrentY = ConvertToPixelsY(Value) End Set End Property

Public Function ConvertToPixelsX(ByVal num As Double) As Double Return IIf(ScaleMode = ScaleModeConstants.vbTwips, _ VB6.TwipsToPixelsX(num), num)

End Function

Public Function ConvertToPixelsY(ByVal num As Return IIf(ScaleMode = ScaleModeConstants VB6.TwipsToPixelsY(num), num)

End Function End Class

Tables 7.3 and 7.4 list Visual Basic 6.0 Printer object properties and methods and their Visual Basic .NET equivalents. Where there are no direct equivalents, links are provided to additional information.

The following example shows you how to convert Visual Basic 6.0 code that uses a printer object to Visual Basic .NET code that uses your own printer class as mentioned earlier.

Printer.FillColor = vbBlue Printer.FillStyle = vbSolid Printer.Circle (3000, 3000), 1500, vbRed Printer.EndDoc

The preceding Visual Basic 6.0 code will print a circle filled with blue color and it will have a red border. To convert this code to Visual Basic .NET, you can use the printer class mentioned earlier and perform the following steps.

First, you have to add this printer class to the .NET project. After this, you have to add a reference to the Micosoft.VisualBasic.Compatibility assembly.

The next step is to create a Visual Basic .NET module and add the following code.

Module Module1

Public Printer As PrinterClass = New PrinterClass End Module

Finally, you need to perform some small changes to the original Visual Basic 6.0 code to make it consistent with the new definition of your own printer class. Here is the resulting Visual Basic .NET code.

Printer.FillColor = Color.Blue

Printer.FillStyle = PrinterClass.FillStyleConstants.vbFSSolid Printer.Circle(New Point(3000, 3000), 1500, Color.Red) Printer.EndDoc()

Double) As Double .vbTwips, _

Table 7.3: Visual Basic .NET Equivalents for Printer Object Properties Visual Basic 6.0 Visual Basic .NET Equivalent

	PrintDocument.PrinterSettings.DefaultPageSettings.Color
ColorMode	
Copies	PrintDocument.PrinterSettings.Copies
CurrentX	No equivalent. Replaced by location and dimension arguments of various

	methods of the Graphics class.This property could be simulated in your printer class with a double variable.
CurrentY	No equivalent. Replaced by location and dimension arguments of various methods of the Graphics class.This property could be simulated in your printer class with a double variable.
DeviceName	PrintDocument.PrinterSettings.PrinterName
DrawMode	No equivalent. For details, see "Graphics Changes in Visual Basic .NET" on MSDN.
DrawStyle	System.Drawing.Drawing2D.DashStyle
DrawWidth	System.Drawing.Pen.Width
DriverName	No equivalent. No longer needed; printer drivers are managed by Windows.
Duplex	System.Drawing.Printing.Duplex
FillColor	No equivalent. For details, see "Graphics Changes in Visual Basic .NET" on MSDN.This property could be simulated in your printer class with a System.Drawing.Color variable.
FillStyle	This functionality can be obtained using a Drawing.SolidBrush to fill the Draws objects.
Font	This property could be simulated in your printer class with a System.Drawing.Font class.
FontBold	To get the value, use: System.Drawing.Font.Bold.To set the value, use: VB6.FontChangeBold.
FontCount	System.Drawing.FontFamily.GetFamilies().Length
FontItalic	To get the value, use: System.Drawing.Font.Italic.To set the value, use: VB6.FontChangeItalic.
FontName	To get the value, use: System.Drawing.Font.Name.To set the value, use: VB6.FontChangeName.
Fonts	System.Drawing.FontFamily.GetFamilies
FontSize	To get the value, use: System.Drawing.Font.Size.To set the value, use: VB6.FontChangeSize.
FontStrikethru	To get the value, use: System.Drawing.Font.Strikeout.To set the value, use: VB6.FontChangeStrikeout.

continued continued

Visual Basic 6.0 Visual Basic .NET Equivalent	
	No equivalent. For details, see "Font Changes in Visual Basic .NET" in Visual Basic Concepts on MSDN.
FontTransparent	
FontUnderline	To get the value, use: System.Drawing.Font.Underline.To set the value, use: VB6.FontChangeUnderline.
ForeColor	System.Drawing.Pen.Color
hDC	PrintDocument.PrinterSettings.GetHdevmode.ToInt32
Height	PrintDocument.DefaultPageSettings.PaperSize.Height
Orientation	PrintDocument.DefaultPageSettings.Landscape
Page	No direct equivalent. The current page number is not tracked; however, you can easily do this by setting a variable in the BeginPrint event and incrementing it in the PrintPage event.
PaperBin	PrintDocument.PrinterSettings.PaperSources
PaperSize	PrintDocument.DefaultPageSettings.PaperSize
Port	No longer necessary. The PrintPreviewDialog control automatically sets port information.
PrintQuality	PrintDocument.DefaultPageSettings.PrinterResolution
RightToLeft	No longer necessary. The direction of printing is controlled by the localization settings in Windows.
ScaleHeight	No equivalent. For details, see "Coordinate System Changes in Visual Basic .NET" in Visual Basic Concepts on MSDN.
ScaleLeft	No equivalent. For details, see "Coordinate System Changes in Visual Basic .NET" on MSDN.
ScaleMode	No equivalent. For details, see "Coordinate System Changes in Visual Basic .NET" on MSDN.
ScaleTop	No equivalent. For details, see "Coordinate System Changes in Visual Basic .NET" on MSDN.
ScaleWidth	No equivalent. For details, see "Coordinate System Changes in Visual Basic .NET" on

	MSDN.
TrackDefault	No direct equivalent. The IsDefaultPrinter property of the PrinterSettings class can be used to determine whether a printer is the default, but printing is no longer halted if the default printer changes.
TwipsPerPixelX	No longer necessary. Measurements in Visual Basic .NET are always in pixels.
TwipsPerPixelY	No longer necessary. Measurements in Visual Basic .NET are always in pixels.
Width	PrintDocument.DefaultPageSettings.PaperSize.Height
Zoom	No longer necessary. If the printer has zoom capabilities, settings are automatically exposed in the Print dialog box.

Table 7.4: Visual Basic .NET equivalents for Printer object methods

Visual Basic 6.0	Visual Basic .NET Equivalent
Circle	PrintPageEvents.Graphics.DrawEllipse
EndDoc	PrintDocument.Print
KillDoc	PrintEventArgs.Cancel
Line	PrintPageEvents.Graphics.DrawLine
NewPage	This method needs to be simulated with an array of pages and call the method Print of the class PrintDocument for each page.
PaintPicture	PrintPageEvents.Graphics.DrawImage
Print	PrintPageEvents.Graphics.DrawString
PSet	PrintPageEvents.Graphics.DrawLine
Scale	No equivalent. For details, see "Coordinate System Changes in Visual Basic .NET" in Visual Basic Concepts on MSDN.
ScaleX	No equivalent. For details, see "Coordinate System Changes in Visual Basic .NET" on MSDN.
ScaleY	No equivalent. For details, see "Coordinate System Changes in Visual Basic .NET" on MSDN.
TextHeight	System.Drawing.Graphics.MeasureString
TextWidth	System.Drawing.Graphics.MeasureString

Upgrading the Printers Collection

Visual Basic 6.0 provides the Printers collection, which is used to return information about available printers on a system. For example, it is possible to determine the printer layout for a particular printer. Visual Basic .NET does not support the Printers collection. The printing model has changed, and as a result, references to the Printers collection require manual adjustment when upgrading to Visual Basic .NET.

Consider the following Visual Basic 6.0 example, which uses the Printers collection to find the first available printer with its page orientation set to Portrait. If such a printer is found in the collection, a message box showing the printer device name is displayed. The code for the example follows.

```
Dim prn As Printer For Each prn In Printers

If prn.Orientation = vbPRORPortrait Then MsgBox prn.DeviceName ' Stop looking for a printer. Exit For End If
Next
```

Because the Printers collection is not supported in Visual Basic .NET, this code cannot be automatically upgraded with the upgrade wizard. Applying the upgrade wizard results in the code being marked with upgrade issues, such as the one demonstrated here.

```
' UPGRADE_ISSUE: Printers collection was not upgraded. For Each prn In Printers
```

The only way to achieve a similar effect in Visual Basic .NET is to implement your own Printers collection, using various classes the Microsoft .NET Framework provides. For example, you can create a new Visual Basic .NET module and class like the following.

```
Module PrintersModule

Public Printers As New PrintersCollection End Module
```

```
Public Class PrintersCollection : Implements IEnumerable

Private printer As System.Drawing.Printing.PrinterSettings

Public ReadOnly Property Count() As Integer Get

Return System.Drawing.Printing.PrinterSettings.InstalledPrinters.Count End Get End Property

Public ReadOnly Property Item(ByVal Index As Integer) _

As System.Drawing.Printing.PrinterSettings

Get printer = New System.Drawing.Printing.PrinterSettings printer.PrinterName = _

System.Drawing.Printing.PrinterSettings.InstalledPrinters.Item(Index) Return printer End Get End Property

Overridable Function GetEnumerator() As IEnumerator _

Implements IEnumerable.GetEnumerator

Dim Count As Integer = _

System.Drawing.Printing.PrinterSettings.InstalledPrinters.Count Dim printersArray(Count) As
System.Drawing.Printing.PrinterSettings Dim i As Integer For i = 0 To Count - 1

printersArray(i) = New System.Drawing.Printing.PrinterSettings printersArray(i).PrinterName = _

System.Drawing.Printing.PrinterSettings.InstalledPrinters.Item(i)

Next

Return printersArray.GetEnumerator() End Function End Class
```

After the preceding module and class are created and added to the upgrade solution, the original code can be upgraded with only a few minor tweaks. These are shown here (changes are highlighted in bold).

```
Dim prn As System.Drawing.Printing.PrinterSettings

For Each prn In Printers

If prn.DefaultPageSettings.Landscape = False Then MsgBox(prn.PrinterName)

Exit For End If Next
```

For more information about replacing the Printers collection functionality in Visual Basic .NET, see "Printers Collection Changes in Visual Basic .NET" in Visual Basic Concepts on MSDN.

Continue reading here: Upgrading the Clipboard Object

- ownload source code - 62.8 KB

Introduction

In this article, I will explain how to communicate with **PLC** (**Programmable Logic Controller**). I started my PLC communication program using .NET from 2007.

For a basic understanding of what PLC is, use Google because basically I am not a PLC Engineer or Electrical Engineer. However, I will explain to you how to connect PLC using .NET programs, how to read data from PLC, and how to write data to PLC. My sample programs are all based on MELSEC PLC using TCP/IP communication.

There are different kinds of PLC available like MELSEC ,SIMENS, etc. For each PLC communication, there are different types of protocols available. In this demo program, I have used TCP IP communication for MELSEC PLC.

Here, I have mentioned .NET programs instead of a language in .NET. In my article, you will find four zip files:

SPLCSharpConnect.zip contains the PLC component DLL source code. I started PLC communication programming in 2007. At that time, I was coding using VB.NET. I created my own Winsock component using VB.NET. Even now for PLC communication, I am using the same Winsock component *MelsecPLC.dll* in my projects. In that component, I have created simple functions as `Connect`, `Disconnect`, `Read`, `Write` and a `DataArrival` event. In the sample project attached, I have used the same DLL.

Before we start PLC communication programming, I'd like to explain the read and write processes in PLC communication.

Read Data from PLC

PLC Read Command: To read data from PLC, we need to send the command to PLC. Basically, the command we send will be like this: "`500000FF03FF000018000A04010000D*0095000001`";

C#

```
String   cmd = "";
cmd = cmd + "5000" ;  // sub HEAD (NOT)
cmd = cmd + "00"  ;   //   network number (NOT)
cmd = cmd + "FF"  ;   //PLC NUMBER
cmd = cmd + "03FF" ;  // DEMAND OBJECT MUDULE I/O NUMBER
cmd = cmd + "00" ;    //  DEMAND OBJECT MUDULE DEVICE NUMBER
cmd = cmd + "001C"  ; //  Length of demand data
cmd = cmd + "000A";   //  CPU inspector data
cmd = cmd + "0401";   //  Read command (to read the data from PLC we should "0401"
cmd = cmd + "0000" ;  //  Sub command
cmd = cmd +"D*"  ;    //   device code
cmd = cmd + "009500"; //adBase
cmd = cmd + "0001";
//Device No ,It’s a Address every PLC device will have an address
//we need to send the appropriate address to read the data.
```

Write Data to PLC

PLC Write Command: To write data to PLC, we need to send the command to PLC. Basically, the command we send will be like this: "`500000FF03FF00001C000A14010000D*0095010002`".

C#

```
String   cmd = "";

cmd = cmd + "5000";   // sub HEAD (NOT)
cmd = cmd + "00";     // network number (NOT)
cmd = cmd + "FF";     // PLC NUMBER
cmd = cmd + "03FF";   // DEMAND OBJECT MUDULE I/O NUMBER
cmd = cmd + "00";     // DEMAND OBJECT MUDULE DEVICE NUMBER
cmd = cmd + "001C";   // Length of demand data
cmd = cmd + "000A";   // CPU inspector data
cmd = cmd + "1401";   // Write command
```

```
cmd = cmd + "0000";   // Sub command
cmd = cmd + "D*";     // device code
cmd = cmd + "009501"; // adBase
cmd = cmd + "0002";   // BASE ADDRESS
```

You can see the difference, to read we use "`0401`" and "`009500`" but for write we use "`1401`" and "`009501`". The detailed code will be listed below.

Using the Code

I have attached C# sample programs for PLC communication in this article but for explanation here, I have used my component *MelsecPLC.dll* in the test project. First, add *MelsecPLC.dll* to your project.

1. Declare MelsecPLC.dll in the Form

C#

```
//
using MelsecPLC;
```

2. Variable Declarations

C#

```
//
public MelsecPLC.Winsock winsock1; // Declare the MelsecPLC Winsock Component
string Jig01;                      // To store the PLC read data.
```

3. Connect (PLC Connection)

For connection, we need the PLC IP address. Here, my PLC IP address is "`10.126.224.221`". Enter the local port and remote port. Here, I have used "1027" for the local port and "8000" for the remote port. In form load, we call the PLC `Connect` function and we create a MelsecPLC `DataArrival` event. The `DataArrival` event will be triggered when PLC sends data to the PC (our computer).

The detailed connect functions and `DataArrival` declaration in the form load are listed below:

C#

Shrink ▲

```
//
private void Form1_Load(object sender, EventArgs e)
{
    winsock1Connect();
    winsock1.DataArrival +=
      new MelsecPLC.Winsock.DataArrivalEventHandler(winsock1_DataArrival);
    timer1.Enabled = true;
    timer1.Start();
}

private void winsock1Connect()
{
    try
    {
        if (winsock1.GetState.ToString() != "Connected")
        {
            winsock1.LocalPort = 1027;
            winsock1.RemoteIP ="10.126.224.221";
            int a = 8000;
            winsock1.RemotePort = 8000;
            winsock1.Connect();
        }
    }
    catch (Exception ex)
    {
```

```
    }
}
```

4. DataArrival Event

In the `DataArrival` event, we get the actual data sent by PLC after the Read signal is sent to PLC. This event will be triggered whenever PLC sends data to the PC. To get data from PLC, we use the `winsock1.GetData()` method.

C#

```
//
private void winsock1_DataArrival(MelsecPLC.Winsock sender, int BytesTotal)
{
    String s = String.Empty;
    winsock1.GetData(ref s);
    Jig01 = s;
}
```

5. Read Data from PLC

To read data from PLC, we need to send the signal to PLC for data read. I have explained the read functions in my article introduction. To send read signal to PLC, we use the `winsock1.Send()` method.

C#

Shrink ▲

```
//
private void btnRead_Click(object sender, EventArgs e)
{
    if (winsock1.GetState.ToString() != "Connected")
    {
        winsock1Connect();
    }
    //String cmd = "500000FF03FF000018000A04010000D*0095000001";
    String cmd = "";
    String OutAddress = "0001";
    cmd = "";
    cmd = cmd + "5000" ;    // sub HEAD (NOT)
    cmd = cmd + "00"  ;     // network number (NOT)
    cmd = cmd + "FF"  ;     // PLC NUMBER
    cmd = cmd + "03FF"   ;  // DEMAND OBJECT MUDULE I/O NUMBER
    cmd = cmd + "00"  ;     // DEMAND OBJECT MUDULE DEVICE NUMBER
    cmd = cmd + "001C"  ;   // Length of demand data
    cmd = cmd + "000A";     // CPU inspector data
    cmd = cmd + "0401";     // Read command
    cmd = cmd + "0000" ;    // Sub command
    cmd = cmd +"D*"  ;      // device code
    cmd = cmd + "009500";   // adBase
    cmd = cmd + OutAddress; // BASE ADDRESS
    winsock1.Send(cmd);
}
```

6. Write Data to PLC

To write data to PLC, we need to send the signal to PLC for data write. I have explained the `write` functions in my article introduction. To send a write signal to PLC, here we use the `winsock1.Send()` method.

C#

Shrink ▲

```
//
private void btnWrite_Click(object sender, EventArgs e)
```

```
{
    if (winsock1.GetState.ToString() != "Connected")
    {
        winsock1Connect();
    }
    String cmd = "";
    String OutAddress = txtWrite.Text.Trim();
    cmd = "";
    cmd = cmd + "5000";      // sub HEAD (NOT)
    cmd = cmd + "00";        // network number (NOT)
    cmd = cmd + "FF";        // PLC NUMBER
    cmd = cmd + "03FF";      // DEMAND OBJECT MUDULE I/O NUMBER
    cmd = cmd + "00";        // DEMAND OBJECT MUDULE DEVICE NUMBER
    cmd = cmd + "001C";      // Length of demand data
    cmd = cmd + "000A";      // CPU inspector data
    cmd = cmd + "1401";      // Write command
    cmd = cmd + "0000";      // Sub command
    cmd = cmd + "D*";        // device code
    cmd = cmd + "009501";    // adBase
    cmd = cmd + OutAddress;  // BASE ADDRESS
    winsock1.Send(cmd);
}
```